(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
22 March 2001 (22.03.2001)

PCT

(10) International Publication Number
WO 01/20918 A2

(51) International Patent Classification[7]: H04Q

(21) International Application Number: PCT/US00/25540

(22) International Filing Date:
18 September 2000 (18.09.2000)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/154,368 17 September 1999 (17.09.1999) US

(71) Applicant: MERCURY INTERACTIVE CORPORATION [US/US]; 1325 Borregas Avenue, Sunnyvale, CA 94089 (US).

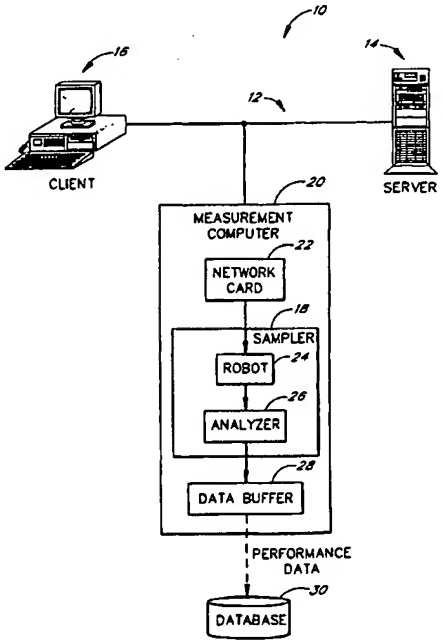
(72) Inventors: RAZ, David; Nes Ziona (IL). COHEN, Ya'Acov; Petach Tikua (IL). AZULAI, Sharon; Sunnyvale, CA (US). KLEIN, Philippe; Jerusalem (IL).

(74) Agent: DELANEY, Karoline, A.; Knobbe, Martens, Olson and Bear, LLP, 16th Floor, 620 Newport Center Drive, Newport Beach, CA 92660 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, CZ (utility model), DE, DE (utility model), DK, DK (utility model), DM, DZ, EE, EE (utility model), ES, FI, FI (utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (utility model), SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

[Continued on next page]

(54) Title: SERVER AND NETWORK PERFORMANCE MONITORING



(57) Abstract: The performance of a client-server network is measured by passively monitoring a TCP packet stream between a client and a server during a communications session. The packet stream may be monitored by a device which is local to the client, by a device which is local to the server, or both. By measuring elapsed times between TCP messages associated with selected events or stages of the session, separate response times and other performance parameters are calculated for the network and for the server, including connect time, network latency, server response times, network bandwidth, and server bandwidth. Because the message stream is monitored at an application-independent protocol layer and without regard to application-dependent events or messages, the method is not limited to specific applications, and does not require advance knowledge of the type of application data requested by the client.

WO 01/20918 A2

(84) **Designated States** *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published:

— *Without international search report and to be republished upon receipt of that report.*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SERVER AND NETWORK PERFORMANCE MONITORING

FIELD OF THE INVENTION

The present invention relates to systems and methods for determining server and network response times and other performance parameters within a client-server network by monitoring communications between a client and a server.

BACKGROUND OF THE INVENTION

Networks, including local area networks (LANs) and wide area networks (WANs), are becoming increasingly prevalent as the number of computational devices in organizations grows. Networks enable information to be shared between computational devices, and as such are important for the ease and convenience of storing and accessing data throughout an organization. Networks are implemented with a connection between at least two computational devices or other network hardware devices. This connection can exist, for example, over a cable or a wireless link, and can be implemented using optical, electrical, infra-red or radio wave based signals (or a combination thereof). Data is passed through this connection according to various protocols at different layers of the network. These protocols include but are not limited to, transmission control protocol (TCP), Internet protocol (IP), Internet packet exchange (IPX), systems network architecture (SNA), datagram delivery protocol (DDP) and so forth. At the data link layer, such protocols include, but are not limited to, Ethernet, token ring, and fiber distributed data interface (FDDI).

Within these different types of protocols and network architectures, a common need exists for monitoring the response time of both servers within the network and the network itself. Since the client-server model is found throughout these different network architectures, the need for such monitoring is not limited to any one particular type of network. Furthermore, the client-server model requires both an efficient server and an efficient network for high performance operation. Thus, pinpointing any problems in the performance of either the network and/or the server is highly important for the operation of the client-server system.

Existing tools and technologies for measuring the performance of a client-server system are deficient in a number of respects. For example, some technologies cannot differentiate between the performance of the server and the performance of the network. In addition, many technologies are specific to particular types of applications, and are therefore are not suitable for monitoring generic client-server sessions. In addition, many technologies are specific to the application layer, and as such, can only monitor performance from the server itself. The present invention addresses these and other problems with the prior art.

SUMMARY OF THE INVENTION

In accordance with the invention, a message stream between a client and a server is monitored from a location on a network in order to determine specific parameters of the performance of the network, the server, or

both. Preferably, the message stream is a TCP or other non-application-specific message stream, and is monitored passively by a computational device that is local to the client; by a device that is local to the server, or both. One or more specific performance parameters are measured by determining the amount of time between specific messages that correspond to specific stages or events of a client-server session. Because these messages and events are not application-dependent, the performance parameters may be determined without regard to the type of application used within the session, and without prior knowledge of the type of application data requested by the client. In a preferred embodiment, the measured performance parameters include connect time, network latency, server response time, network bandwidth, and server bandwidth.

One aspect of the invention is thus a method of monitoring performance of a client-server network from a location on the network. The method comprises (a) monitoring a message stream on the network between a client and a server during a client-server session, (b) detecting a first message of the message stream and a second message of the message stream, wherein the first and second messages are selected to correspond to non-application-dependent events of the session, and (c) measuring an elapsed time between detection of the first message and detection of the second message to determine a parameter of either network performance or server performance. Performance is thereby measured without regard to a type of application used within the client-server session.

Another aspect of the invention is a system for monitoring performance of a client-server network from a location on the network. The system comprises a computational device connected to the network and configured to monitor a message stream between a client and a server at an application-independent protocol level during a client-server session. The system further comprises a program module which uses the message stream as monitored by the computational device to measure performance of the network and/or the server. The program module measures an elapsed time between receipt by the computational device of a first message of the message stream, and receipt by the computational device of a second message of the message stream, to determine a parameter of either network performance or server performance. The first and second messages correspond to selected non-application-dependent events within the client-server session.

Another aspect of the invention is a system for monitoring performance of a client-server network from a location on the network. The system comprises a computer connected to the network and configured to passively monitor a Transmission Control Protocol (TCP) message stream between a client and a server during a session between the client and the server. The system further comprises a program module which uses the message stream as monitored by the computer to measure performance of the network and/or the server. The program module measures an elapsed time between receipt by the computer of a first TCP message of the message stream, and receipt by the computer of a second TCP message of the message stream, to determine a parameter of either network performance or server performance, wherein the first and second TCP messages correspond to selected non-application-dependent events within the session.

BRIEF DESCRIPTION OF THE DRAWINGS

The foregoing and other objects, aspects and advantages will be better understood from the following detailed description of the preferred embodiments of the invention with reference to the drawings, wherein:

FIG. 1 is a schematic block diagram of an illustrative system according to the present invention;

FIG. 2 is a schematic block diagram of an illustrative session for measurement of performance parameters according to the present invention; and

FIG 3 is a flow diagram of a process for measuring performance parameters of the type illustrated in FIG 2.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention provides a system and method for automatic measurement of the performance of a client-server network, including the separate measurement of the performance of the server and of the network. The performance measurements are generated by measuring elapsed times between specific messages between the client and server as monitored from a location at or near the server, and/or from a location at or near the client. As will be recognized, the measurements could be taken from substantially any location on the network from which the client-server message stream may be monitored, although the "meanings" of the measurements are dependent upon the monitoring location.

The method or process of the present invention can be described as a series of steps or tasks implemented by a data processor, such that the process can be implemented as hardware, software or firmware, or a combination thereof. In a preferred embodiment, the process is embodied within a software application (program) which runs on one or more general-purpose computers. The software application may be written in any suitable programming language such as C, C++ and Java.

In accordance with the invention, messages between the client and server are analyzed at the TCP layer, or other application-independent protocol layer, in order to determine the response times and other performance parameters of the server and/or the network. The various response times and other parameters are measured without reliance on application-specific information within the message stream. An important benefit of this method is that no advance knowledge is required of the type of application data requested by the client. The data analysis is optionally and preferably performed by listening for TCP frames that indicate the state of the session between the client and the server. Preferably, the TCP messages are monitored with a hardware connection device connected to the network in a promiscuous mode in which all local network traffic passes through the connection device.

For simple requests from the client for a single type of data, the analysis is performed on data received through a single port. For more complex requests from the client involving multiple types of data, data received through each port is preferably analyzed separately for the response time of the server and/or of the network. Since such complex requests are actually a combination of multiple simple requests, the performance of the server and/or of the network can be determined by decomposition of the received data. In addition, as previously mentioned, separate measurements may be performed at or near the server, and at or near the client on the network.

The principles and operation of a method and system according to the present invention may be better understood with reference to the drawings and the accompanying description, it being understood that these drawings are given for illustrative purposes only and are not meant to be limiting.

Referring now to the drawings, Figure 1 is a schematic block diagram of a system 10 for determining the performance of a network 12. As shown, the system includes a server 14 and a client 16, which are typically located remotely from one another. The client and the server are connected to network 12, which is preferably a network that supports TCP. Although illustrated generally as a PC, the client device 16 may, for example, be a handheld computer, a WAP (Wireless Access Protocol) or other wireless phone, a data appliance, or any other type of data processing device which makes requests on a network as a client. Likewise, the server 14 can be any type of data processing device, or combination of data processing devices, which respond to such requests as a server.

The system also includes a sampler 18 connected to the network 12. Sampler 18 is preferably operated by a computational device or computer 20, also termed a "measurement computational device" for distinguishing computational device 20 from server 14 and the client 16. The measurement computational device 20 includes a network card 22 or other network connector hardware device. The network card 22 is able to listen to local traffic passing through or along the portion of the network 12 to which the measurement computational device is connected. The connection of the measurement computational device 20 to the network 22 is preferable a hardwire connection, but could alternatively be a wireless connection (e.g., in wireless LAN environments). The sampler could alternatively run on the client device 16 or the server device 14 (or both).

The computational device 20 may optionally be located at or near server 14, or at or near client 16. In each case, measurements of network performance may be made. Further, one computational device 20 may be located local to the client 16 and another computational device 20 located local to the server 14, such that a client-server session is monitored from both locations concurrently (to obtain different types of measurements, as described below). Where multiple devices are used to monitor the same session, the resulting performance measurements may be appropriately combined for purposes of analysis and reporting.

In promiscuous mode, the network card 22 receives all packets that pass through or along a portion of network 12, regardless of whether these packets are specifically addressed to network card 22 itself. Setting the network card 22 to operate in promiscuous mode is only one example of a mechanism for eavesdropping on network traffic flowing past the computational device 20 on the network 12. Adapted mechanisms could be used for eavesdropping on network traffic for networks operating through microwave or fiber optic transmissions, for example, which do not operate with network interface cards. Such adaptations could easily be performed by one of ordinary skill in the art.

A robot 24 initiates sessions with server 14, in place of client 16, resulting in the transmission of data, for example by requesting a Web page or any type of mark-up language document or object. Alternatively and preferably, robot 24 is passive, in which case the information obtained by network card 22, which is described in greater detail below, is passed to robot 24. Robot 24 watches for the initiation of such a session between client 16 and server 14,

and then monitors the transmission of data during the session. Because the session is monitored at the TCP layer, the ability to reliably track the session is not dependent upon the particular type of client-server application being used.

The results obtained by robot 24, concerning the various events or stages of the session and the amount of time elapsed between such stages, are passed to a data buffer 28. Robot 24 is optionally and preferably implemented as an application software module, but is alternatively implemented as hardware (such as within a programmable ASIC or other integrated circuit device) or within firmware. Optionally, robot 24 is not used to perform the measurements, which instead are performed by the server 14 and/or the client 16.

An analyzer 26 also receives the information obtained by network card 22. Analyzer 26 calculates the amount of time required to reach each stage, thereby determining the response time of the server 14 and/or of the network 12. As described in greater detail below with regard to the TCP protocol for data transmission, analyzer 26 uses the flags or events for each stage of the session, obtained by robot 24, as well as the elapsed time at the receipt of each flag to determine various response times. Although Figure 2 illustrates the performance of robot 24 and analyzer 26 with regard to the TCP protocol for data transmission, the response times could be similarly measured for other protocols in which communication during a session occurs in clearly demarcated stages.

Analyzer 26 preferably sends the analyzed data to data buffer 28. Data buffer 28 optionally correlates the information between analyzer 26 and robot 24, preferably performing a complete analysis of an entire session for response time, latency, and other parameters. The performance data (response time measurements, etc.) may optionally be collected from a plurality of samplers 18 and stored in a database 30. For example, a plurality of samplers 18 could be deployed at selected client locations (e.g., in different office buildings, cities or countries), and used to monitor performance as seen from such locations; the resulting performance data generated by such samplers may be reported to and aggregated within a central database 30 used to generate online performance reports.

Figure 2 shows an example client-server session, and illustrates the measurements taken by the sampler 18 (or samplers) in a preferred embodiment of the invention. In the illustrated embodiment, the client and the server exchange packets with TCP commands or fields according to the TCP protocol. As will be apparent, the specific performance parameters measured during the session depend upon whether a sampler is located local to the client, local to the server, or both. Specific steps or events within the session are labeled with numbers. Measurements shown on the client side of the drawing are taken using a computational device 20 that is local to the client, such that the sampler receives a packet at substantially the same time it is transmitted or received by the client. Measurements shown on the server side of the drawing are taken using a computational device 20 local to the server, such that the sampler receives a packet at substantially the same time it is transmitted or received by the server.

At event 1, the client sends a synchronize (SYN) request to the server. At event 2, the server replies to the client with a synchronize acknowledgement (SYN/ACK) message, both acknowledging receipt of the SYN request and sending its own SYN request. The elapsed time between transmission of the SYN request by the client and the receipt of the SYN/ACK reply by the client is the connect time (a specific network latency measurement), when such connect time is measured at or near the client.

At event 3, the client sends an ACK message to the server. If the measurements are taken at or near the server, the connect time may be measured as the elapsed time between transmission of the SYN/ACK reply and receipt of another ACK message (labeled "server connect time" in Figure 2).

At event 4, which occurs shortly after event 3, the client sends a request to the server for particular data, such as a GET command message. The server then replies to the client with an ACK message at event 5. Since the server sends such a reply without any processing of the request and without regard to the type of application which must handle the request, the amount of server processing time (referred to as the "kernel time") required to send the ACK message is normally very low.

The amount of time that elapses between transmission of the GET message or other data request command by the client, and receipt of the ACK message from the server, as measured at or near the client, is a measure of the network latency. Such a measurement of network latency may optionally be further refined by subtracting the kernel time of the server, to obtain a more precise measurement of the network response time as measured at or near the client. As previously described, network latency is also optionally measured at or near the server.

At event 6, the server sends the requested data to the client in one or more data response ("R") messages (four R messages shown). The elapsed time between receipt of the GET message by the server and transmission of the first "R" message by the server is the server response time or server latency, as measured at the server. Preparing the requested data may require significant processing time by the server. If the request is simple, for a single type of data for example, then only one response message may be sent.

As further shown in Figure 2, the server's response time may also be approximated by measuring, for a location local to the client, the elapsed time between receipt of the ACK message from the server and receipt of the first R message from the server. This approximation is based on the assumption that the network latency is approximately same for both of these messages.

As illustrated, a sequence of multiple consecutive R (data response) messages may be sent during event 6 to transfer the requested data. After some predetermined number of R messages have been sent, the server waits for, and the client sends, an ACK message (event 7). Upon receiving this ACK message, the server resumes transmissions of response messages as needed. This entire cycle is then repeated (as shown) until all of the data for the response has been sent from the server to the client. As illustrated, the elapsed time (as measured local to the server) between server transmission of the last R message of the sequence and receipt of the corresponding ACK message is a measure of network latency.

As illustrated in Figure 2, another measurement that may optionally be taken is the time between consecutive R (data response) messages. This elapsed period of time is a measure of the time delay caused by the limited availability of bandwidth. If measured near the client, this time period can be used to assess network bandwidth. If measured near the server, this time period can be used to assess server bandwidth.

Each time the server responds to another GET message from the client, the server response time is preferably measured as described above. Additionally and/or alternatively, each time data is sent to a different port,

the server response time may be measured separately as for a separate response.

An additional type of time measurement that may be performed at the server side is from receipt of the ACK message for the last R message to server transmission of a new R message to the client. This elapsed period of time, labeled "resumed transmission response time" in Figure 2, is an additional measure of server performance. Additional measurements of network and server bandwidth, network latency, and resumed transmission response time, may be taken at later stages of the session.

Two other performance parameters that may be determined during the client-server session are the number of server retries and the number of network retries. A server retry occurs when the server does not respond to a message from a client, such that the client must send the message again (not shown in Figure 2). Similarly, a network retry occurs when the client does not send an ACK message to the server, such that the server must send the message again. If the measurements are taken at or near the server, the sampler can distinguish between network retries and server retries by determining whether a packet was dropped by the server versus the network. This is in contrast to prior systems, which generally cannot distinguish between network retries and server retries.

After the transfer is complete, the server sends a FIN message to the client (not shown). The difference between the time that the last response is received by the client and the time that the FIN is received by the client is another measure of network latency, although with some server processing time added. In response to the FIN, the client sends a FIN/ACK message to the server (not shown), and the server then returns an ACK message to the client (not shown). The sequence of FIN (from the server); FIN/ACK (from the client); and ACK (from the server), can be used to determine that a particular data request session is finished and a new session is about to begin, in the case of complex data requests.

Figure 3 illustrates a basic process that may be used by the sampler 18 to monitor performance as described above. The sampler initially initiates, or detects the initiation of, a client-server session. Once the session is initiated, the sampler logs some or all of the TCP messages of the session, including the message type and the time of receipt. Other types of parameters such as the message size may also be recorded. In addition, the stages of the session may be tracked (e.g., via a TCP state machine) as the messages are received (not shown).

Once the session ends or times out, the message log is processed (preferably by the computational device 20, but alternatively by a separate computational device) to identify the various events or stages of the session (if not previously identified) and the associated response times. This analysis task may alternatively be performed in-whole or in-part in real time during session execution. In addition to the response times, network retry events may be counted as indicated above. The type of data analysis performed may be selected based on the location of the sampler 18 relative to the client and the server, as described above.

Once all of the performance measurements have been calculated, the performance data may be uploaded to a local or remote database 30 or otherwise made available to system administrators. If multiple samplers are used to monitor the same session (e.g., one client-side and one server-side sampler), the performance data from such samplers may be correlated and combined for reporting purposes.

The various performance measurements may be incorporated into one or more computer-generated reports made available to system administrators. In addition to the specific measurements discussed above, these reports may include such data as the average, maximum, and total network latency, and the average, maximum, and total server time. These values may be generated over a single session or over multiple sessions. The reports may also include scores indicating the levels of server bandwidth and network bandwidth detected. The various performance measurements may also be used to generate real time alert messages when performance problems are detected.

Thus, the system and method of the present invention provide a simple mechanism for automatically measuring network server performance, together or separately. Because the performance parameters are determined based on non-application-dependent TCP messages and events such as those mentioned above, the system and method do not require use of a particular type of application, and can be used to monitor performance without regard to application type.

It will be appreciated that the above descriptions are intended only to serve as examples, and that many other embodiments are possible within the spirit and the scope of the present invention. The scope of the invention is defined only by the following claims.

WHAT IS CLAIMED IS:

1. A method of monitoring performance of a client-server network from a location on the network, the method comprising:

(a) monitoring a message stream on the network between a client and a server during a client-server session;

(b) detecting a first message of the message stream and a second non-application-dependent message of the message stream, wherein the first and second messages are selected to correspond to non-application-dependent events of the session; and

(c) measuring an elapsed time between detection of the first message and detection of the second message to determine a parameter of either network performance or server performance;

whereby performance is measured without regard to a type of application used within the client-server session.

2. The method as in Claim 1, wherein (a) comprises monitoring the message stream passively.

3. The method as in Claim 1, wherein (a) comprises monitoring the message stream passively using a network card configured in a promiscuous mode.

4. The method as in Claim 1, wherein (a) comprises monitoring the message stream at a network location which is local to the client and remote from the server.

5. The method as in Claim 1, wherein (a) comprises monitoring the message stream at a network location which is local to the server and remote from the client.

6. The method as in Claim 1, wherein the message stream is a Transmission Control Protocol message stream.

7. The method as in Claim 1, wherein the first message is a synchronization message from the client, and the second message is a synchronization acknowledgement message from the server, and wherein the first and second messages are detected at a network location local to the client such that the elapsed time represents a client connect time.

8. The method as in Claim 1, wherein the first message is a data request message from the client, and the second message is an acknowledgement message sent by the server in response to the data request message, and wherein the first and second messages are detected at a network location local to the client such that the elapsed time represents network latency.

9. The method as in Claim 1, wherein the first message is an acknowledgement message received from the server in response to a data request message, and the second message is a first data response message to the data request message, and wherein the first and second messages are detected at a network location local to the client such that the elapsed time represents an approximate server response time.

10. The method as in Claim 1, wherein the first message is a data response message of a sequence of consecutive data response messages transmitted by the server, and the second message is an immediately following data response message of the sequence, and wherein the first and second messages are detected at a network location local to the client such that the elapsed time represents a measure of network bandwidth.

11. The method as in Claim 1, wherein the first message is a data response message of a sequence of consecutive data response messages transmitted by the server, and the second message is an immediately following data response message of the sequence, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents a measure of server bandwidth.

12. The method as in Claim 1, wherein the first message is a synchronization acknowledgement message from the server, and the second message is an acknowledgement from the client to the synchronization acknowledgement message, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents a server connect time.

13. The method as in Claim 1, wherein the first message is a data request message from the client, and the second message is an acknowledgement from the server to the data request message, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents a kernel time for the server.

14. The method as in Claim 1, wherein the first message is a data request message from the client, and the second message is a data response to the data request message, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents a server response time.

15. The method as in Claim 1, wherein the first message is a last message of a sequence of data response messages from the server, and the second message is an acknowledgement from the client of receipt of said last message, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents network latency.

16. The method as in Claim 1, wherein the first message is a client acknowledgement of a last data response message of a sequence, and the second message is a first data response message from the server of an immediately following sequence, and wherein the first and second messages are detected at a network location local to the server such that the elapsed time represents a resumed transmission server response time.

17. The method as in Claim 1, further comprising repeating (b) and (c) at least once to generate multiple server response times, and combining the multiple server response times to generate a total server response time.

18. The method as in Claim 1, further comprising repeating (b) and (c) at least once to generate multiple network response times, and combining the multiple network response times to generate a total or an average network response time.

19. The method as in Claim 1, further comprising repeating (b) and (c) at least once to generate at least one network response time and at least one server response time.

20. The method as in Claim 1, wherein (a), (b) and (c) are performed separately from a location local to the client and a location local to the server to monitor the same session.

21. A system for monitoring performance of a client-server network from a location on the network, the system comprising:

a computational device connected to the network and configured to monitor a message stream between a client and a server at an application-independent protocol level during a client-server session; and

a program module which uses the message stream as monitored by the computational device to measure performance of the network and/or the server;

wherein the program module measures an elapsed time between receipt by the computational device of a first message of the message stream, and receipt by the computational device of a second message of the message stream, to determine a parameter of either network performance or server performance, said first and second messages corresponding to selected non-application-dependent events within the client-server session.

22. The system as in Claim 21, wherein the computational device is configured to monitor the message stream passively.

23. The system as in Claim 21, wherein the computational device monitors the message stream passively using a network card configured in a promiscuous mode.

24. The system as in Claim 21, wherein the computational device is connected to the network locally to the client and remotely from the server.

25. The system as in Claim 21, wherein the computational device is connected to the network locally to the server and remotely from the client.

26. The system as in Claim 21, wherein the message stream is a Transmission Control Protocol message stream.

27. The system as in Claim 21, wherein the first message is a synchronization message from the client, and the second message is a synchronization acknowledgement message from the server, and wherein the computational device is local to the client such that the elapsed time represents a client connect time.

28. The system as in Claim 21, wherein the first message is a data request message from the client, and the second message is an acknowledgement message sent by the server in response to the data request message, and wherein the computational device is local to the client such that the elapsed time represents network latency.

29. The system as in Claim 21, wherein the first message is an acknowledgement message received from the server in response to a data request message, and the second message is a first data response message to the data request message, and wherein the computational device is local to the client such that the elapsed time

represents an approximate server response time.

30. The system as in Claim 21, wherein the first message is a data response message of a sequence of consecutive data response messages transmitted by the server, and the second message is an immediately following data response message of the sequence, and wherein the computational device is local to the client such that the elapsed time represents a measure of network bandwidth.

31. The system as in Claim 21, wherein the first message is a data response message of a sequence of consecutive data response messages transmitted by the server, and the second message is an immediately following data response message of the sequence, and wherein the computational device is local to the server such that the elapsed time represents a measure of server bandwidth.

32. The system as in Claim 21, wherein the first message is a synchronization acknowledgement message from the server, and the second message is an acknowledgement from the client to the synchronization acknowledgement message, and wherein the computational device is local to the server such that the elapsed time represents a server connect time.

33. The system as in Claim 21, wherein the first message is a data request message from the client, and the second message is an acknowledgement from the server to the data request message, and wherein the computational device is local to the server such that the elapsed time represents a kernel time for the server.

34. The system as in Claim 21, wherein the first message is a data request message from the client, and the second message is a data response to the data request message, and wherein the computational device is local to the server such that the elapsed time represents a server response time.

35. The system as in Claim 21, wherein the first message is a last message of a sequence of data response messages from the server, and the second message is an acknowledgement from the client of receipt of said last message, and wherein the computational device is local to the server such that the elapsed time represents network latency.

36. The system as in Claim 21, wherein the first message is a client acknowledgement of a last data response message of a sequence, and the second message is a first data response message from the server of an immediately following sequence, and wherein the computational device is local to the server such that the elapsed time represents a resumed transmission server response time.

37. The system as in Claim 21, wherein the elapsed time represents network latency.

38. The system as in Claim 21, wherein the elapsed time represents a response time of the server.

39. A system for monitoring performance of a client-server network from a location on the network, the system comprising:

a computer connected to the network and configured to passively monitor a Transmission Control Protocol (TCP) message stream between a client and a server during a session between the client and the server; and

a program module which uses the message stream as monitored by the computer to measure performance of the network and/or the server;

wherein the program module measures an elapsed time between receipt by the computer of a first TCP message of the message stream, and receipt by the computer of a second TCP message of the message stream, to determine a parameter of either network performance or server performance, said first and second TCP messages corresponding to selected non-application-dependent events within the session.

40. The system as in Claim 39, wherein the elapsed time represents network latency.

41. The system as in Claim 39, wherein the elapsed time represents a response time of the server.

42. The system as in Claim 39, wherein the elapsed time represents network bandwidth.

43. The system as in Claim 39, wherein the elapsed time represents server bandwidth.

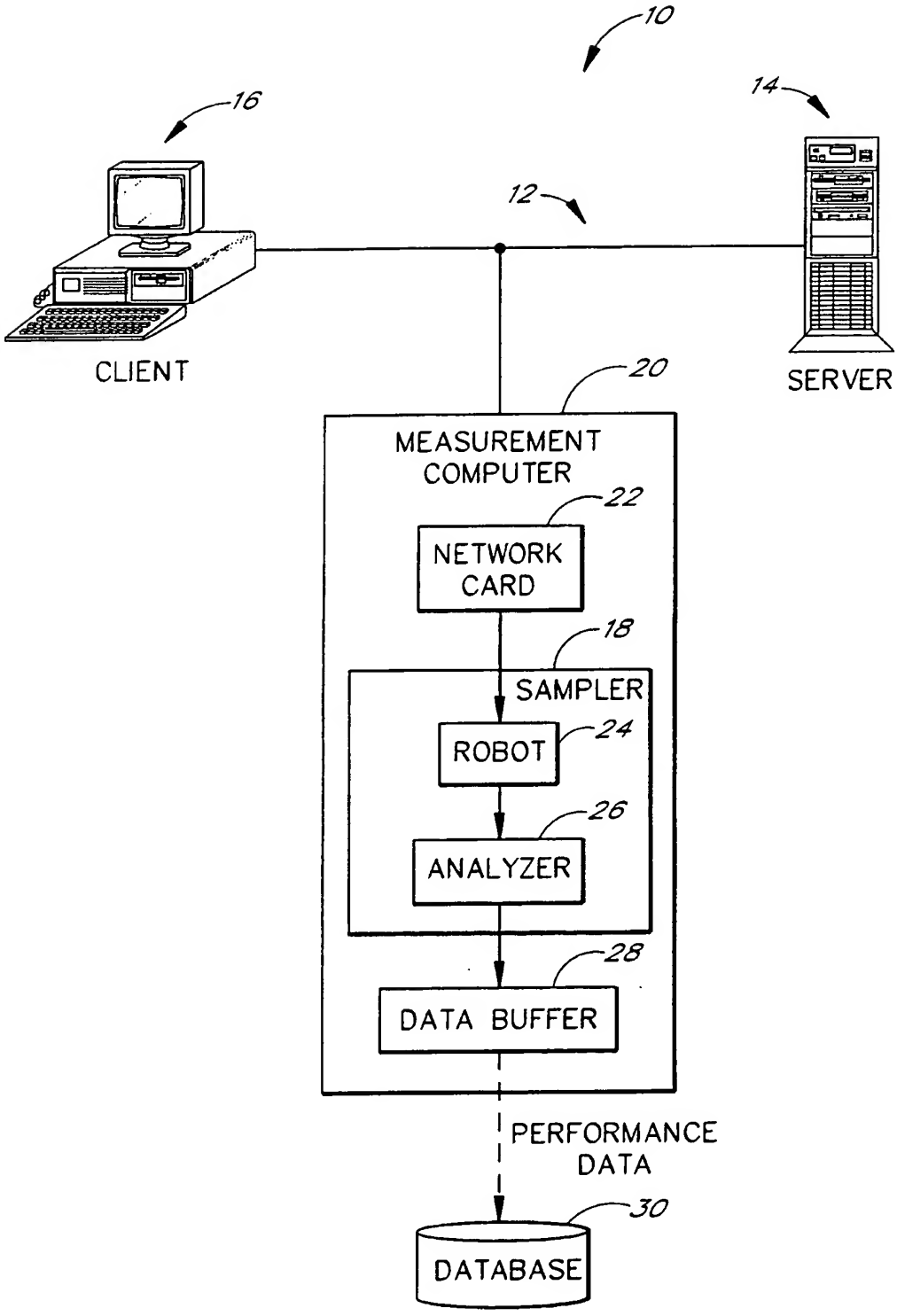

10
16
14
12
CLIENT
SERVER
20
MEASUREMENT COMPUTER
22
NETWORK CARD
18
SAMPLER
ROBOT
24
26
ANALYZER
28
DATA BUFFER
PERFORMANCE DATA
30
DATABASE

FIG. 1

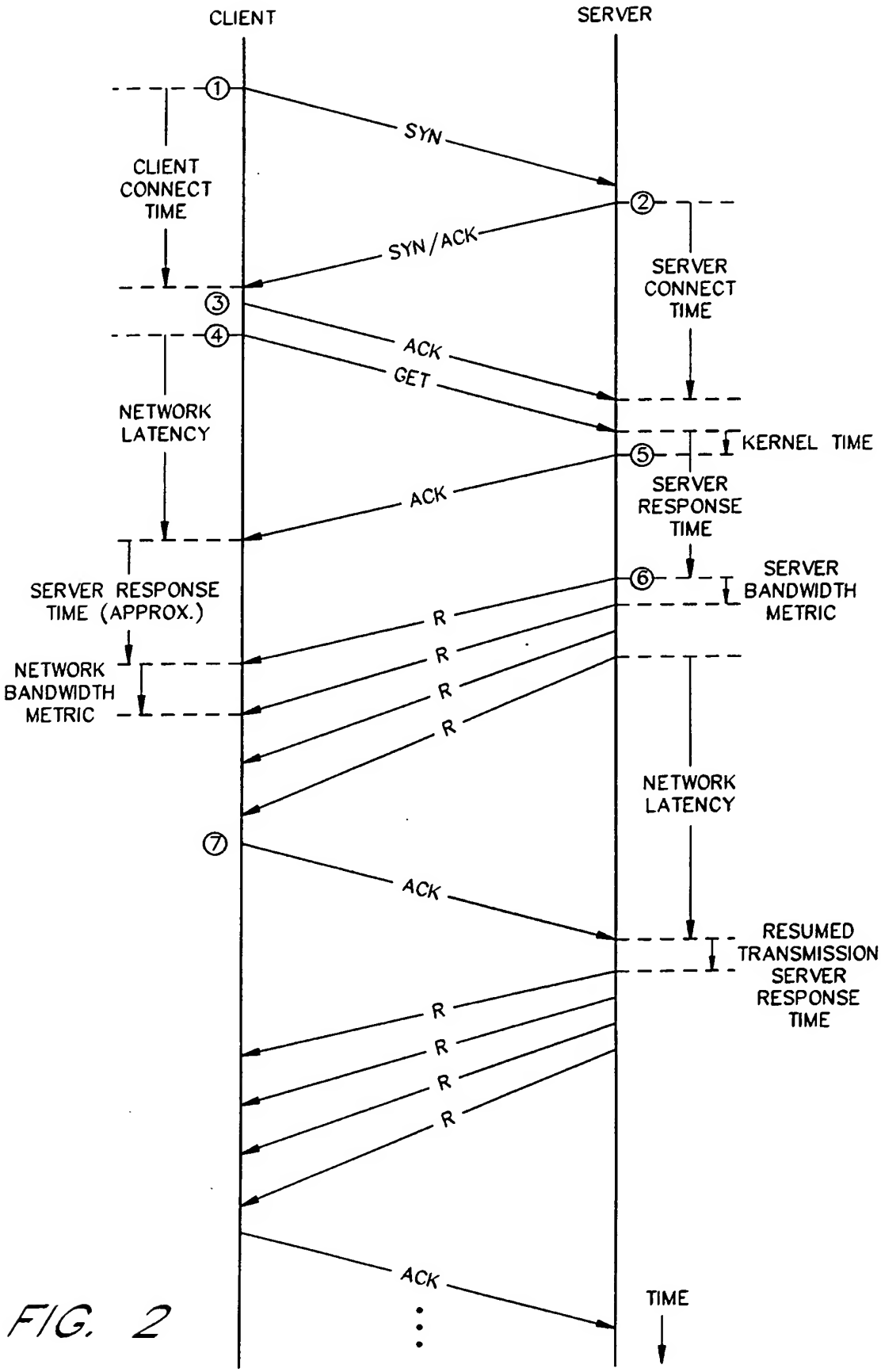

CLIENT
SERVER
1
SYN
CLIENT CONNECT TIME
2
SYN/ACK
SERVER CONNECT TIME
3
4
ACK
GET
NETWORK LATENCY
KERNEL TIME
5
SERVER RESPONSE TIME
ACK
6
SERVER BANDWIDTH METRIC
SERVER RESPONSE TIME (APPROX.)
R
R
R
R
NETWORK BANDWIDTH METRIC
NETWORK LATENCY
7
ACK
RESUMED TRANSMISSION SERVER RESPONSE TIME
R
R
R
R
ACK
TIME

FIG. 2

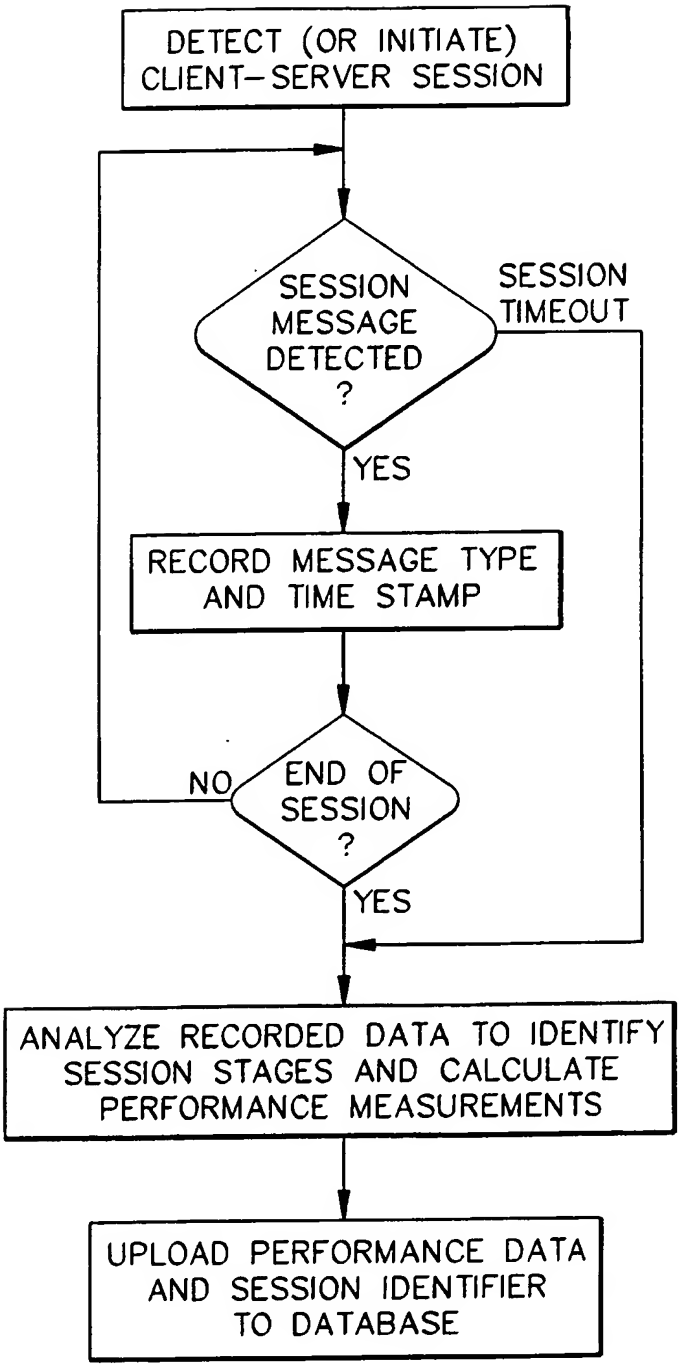

DETECT (OR INITIATE) CLIENT–SERVER SESSION
SESSION MESSAGE DETECTED ?
SESSION TIMEOUT
YES
RECORD MESSAGE TYPE AND TIME STAMP
END OF SESSION ?
NO
YES
ANALYZE RECORDED DATA TO IDENTIFY SESSION STAGES AND CALCULATE PERFORMANCE MEASUREMENTS
UPLOAD PERFORMANCE DATA AND SESSION IDENTIFIER TO DATABASE

FIG. 3